



Art 2161
JFW

PATENT

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

APPLICANT(S): Vandersluis. EXAMINER: Nguyen, Cindy

SERIAL NO.: 09/900,079 ART GROUP: 2161

FILED July 6, 2001 Case No.: XAW-0102

ENTITLED: System and Method for Converting Data in a First Hierarchical Data Scheme into a Second Hierarchical Data Scheme

I hereby certify that this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to: Honorable Commissioner of Patents and Trademarks, P.O. Box 1450, Alexandria, VA 22313-1450 on:

8/24/05
Date of Deposit

Dale B. Halling
Attorney of Record


Signature

APPEAL BRIEF

Honorable Commissioner of
Patents and Trademarks
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This is an appeal from the rejection of claims 1-29 of the Office Action dated March 23, 2005. This application was filed on July 6, 2001. Appellant submits this Appeal Brief pursuant to 35 U.S.C. §134 and 37 C.F.R. § 41.37 in furtherance of the Notice of Appeal filed in this case on May 24, 2005. The fees required under 37 C.F.R. §1.17(b) and any other necessary fees as indicated in the accompanying Appeal Brief Transmittal Letter were paid with the original Appeal Brief filed on May 11, 2005

I. Real Party In Interest

The real party in interest is: XAware Inc., a Colorado corporation, having a place of business at 5555 Tech Center Dr., Suite 200, Colorado Springs, CO 80919. See the Assignment recorded at Reel 012236, Frame 0990.

II. Related Appeals And Interferences

There are no appeals or interferences related to the present appeal.

III. Status Of Claims

Claims 1-29 (see Appendix) are pending in this application. Claims 1-29 are rejected and are involved in this appeal.

IV. Status Of Amendments

There have been no amendments filed subsequent to the rejection of March 23, 2005.

V. Summary Of Claimed Subject Matter

The invention is directed to system and method that provides a simple solution for converting data in a first hierarchical data scheme into a second hierarchical data scheme. A hierarchical data scheme is defined as a scheme that groups data and its context. (Page 2, lines 20-23, specification). Examples of hierarchical data schemes are XML (eXtensible Markup Language) files, relational databases, self describing databases, etc. (Page 2, lines 20-23, specification).

FIG. 2 of the specification is a block diagram of a system 40 for converting data in a first hierarchical data scheme into a second hierarchical data scheme in accordance with one embodiment of the invention. The system 40 has a server 42 that contains a template 44. The template 44 is connected to (or contains) several dynamic data generation modules (DDGM) 46, 48. Note that a DDGM 48 may call another DDGM 50 to complete its task. The DDGMs 46, 48, 50 are connected to a pair of drivers 52, 54. The pair of drivers 52, 54 is connected to a pair of data sources (DS) 56, 58. The template 44 is connected to a client system 60. In one embodiment the client transmits a key and an instruction 62 to the server 42 and receives an XML document related to the key 62. In another embodiment, the client system 60 transmits an XML document 64 such as an order to the server 42 and the order is translated and different portions are sent to the different data sources 56, 58. In yet another embodiment, client system 60 sends a key 62 to the server 42 and receives data 64. Note that the template 44, DDGMs 46, 48, 50 and drivers 52, 54 form a dynamic data conversion program.

The system 40 also includes a developer module 66 connected to the server 42. The developer module 66 has a static template 68 such as a sample XML document, a static extensible markup language document, an XML schema or an XML document type definition (DTD). A wizard 70 or set of wizards walk a user through the process of converting the static template 68 into a dynamic template 72 that is then published to a server 42. The wizards 42 also generate the DDGMs and the drivers. The system 40 may be used to convert between any two hierarchical data schemes. In one embodiment, when a user wants to convert between two data schemes neither of which are XML an intermediate XML document is created.

Thus the system may be used to convert back office data systems into an XML format, or convert an XML formatted order into the format expected by the back office systems (processing systems) or to synchronize two different data systems. This list is suggestive of the potential applications of present invention but not limiting. One feature that makes the present invention so useful is that the development process is simple and is very similar for converting from back office data sources to XML or from XML to back office systems. Both processes use an XML template and the steps are essentially the same.

VI. Grounds of Rejection to be Reviewed on Appeal

1. Whether claims 1-23 & 25-29 are unpatentable over Sun Microsystems (EP 1126380) in view of Bellwood et al. (US 6401132).
2. Whether claim 24 is unpatentable over Sun Microsystems (EP 1126380) in view of Bellwood et al. (US 6401132) and further in view of Povilus (US 5740425).

VII. Argument

As explained in the "Summary of Invention" section, a key problem the present invention solves is the conversion of data in a first hierarchical data scheme into data in a second hierarchical data scheme. Note that a hierarchical data scheme is defined in the specification as a scheme that groups data and its context. (Page 2, lines 20-23, specification). The applicant is allowed to be his own lexicographer see *ZMI Corp v. Cardiac Resuscitator Corp.* 844 F.2d 1576, 6 USPQ2d 1557, 1560. Examples of hierarchical data schemes are XML files, relational databases, self describing databases, etc. (Page 2, lines 20-23, specification). Converting between hierarchical data schemes is a huge problem for corporations and government entities that have legacy systems. XAware, the owner of this patent, has raised tens of millions of dollars of venture capital and has millions of dollars in sales around the world solving this problem in the manner described in this patent application.

1. Whether claims 1-23 & 25-29 are unpatentable over Sun Microsystems (EP 1126380) in view of Bellwood et al. (US 6401132).

All the independent claims (1, 13, 18 & 26) require two hierarchical data schemes. As a result, all the above referenced claims are grouped. Note that a hierarchical data scheme is defined in the specification as a scheme that groups data and its context. (Page 2, lines 20-23, specification). The applicant is allowed to be his own lexicographer see *ZMI Corp v. Cardiac Resuscitator Corp.* 844 F.2d 1576, 6 USPQ2d 1557, 1560. As a result, HTML (Hyper Text Markup Language) or even XML used as a display language is not a hierarchical data scheme. HTML groups data with its presentation format not its context. The Sun reference is concerned with converting content data and formatting data into a pair of XML documents. (See Abstract Sun reference). Formatting data as used in Sun clearly means the presentation of the data on a screen, like a HTML document. (See Sun Paragraph 0007). The goal of Sun is "to provide an XML representation of a computer readable document containing hard formatting properties which allows an easy amendment of the content as well as the style properties of the XML-

document.” (Sun para. 0013). The goal of the present invention is to convert data in a first hierarchical data scheme into data in a second hierarchical data scheme. When the present application and the cited prior art are looked at as a whole there is no suggestion to create the system described in the applicants' claims, see *Medtronic, Inc. v. Cardiac Pacemakers, Inc.*, 721 F.2d 1563, 220 USPQ 97 (Fed. Cir. 1983). So Sun is not even directed to the same problem as the present application. The addition of Bellwood just further confuses the matter. Bellwood is concerned with transcoders, (See abstract) for processing HTTP requests on the fly. (See Bellwood, Col. 3, lines 57-59). Bellwood states as an example, “a client requests a web page hosed on a web server. Transcoder T(1) is used, for example, to modify the request so that the request can pass through a proxy. Transcoder T(2) is used to retrieve the requested page, and transcoder T(3) is used to edit the document (e.g., to delete an image, to remove text, to inject an advertisement and the like)”. (See Bellwood Col. 3, lines 58-65). Bellwood is concerned with web pages and how to modify them on the fly. He is not concerned with hierarchical data schemes.

Independent claims 1 & 13 require a first hierarchical data scheme and a second hierarchical data scheme. Sun discusses the use of HTML (or similar) files and XML files. HTML is a presentation language. As stated above Sun is using HTML as a presentation language and in fact is using XML as a presentation language not a hierarchical data scheme. The question of obviousness requires that we determine if the references, taken as a whole, would suggest the invention to one of ordinary skill in the art. *Medtronic, Inc. v. Cardiac Pacemakers, Inc.*, 721 F.2d 1563, 220 USPQ 97 (Fed. Cir. 1983). Claims 1 & 13 are clearly allowable over the prior art.

Independent claims 18 & 26 both require a static extensible markup language template. The Examiner points to paragraph 0014-0016 of Sun (Office Action 3/23/05, Page 2, item 2). However, a close reading of this section shows that Sun is just discussing converting HTML presentation formatted data into an XML presentation formatted data. There is no suggestion of a static extensible markup language template any where in Sun. The question of obviousness requires that we determine if the references, taken as a whole, would suggest the invention to one of

ordinary skill in the art. *Medtronic, Inc. v. Cardiac Pacemakers, Inc.*, 721 F.2d 1563, 220 USPQ 97 (Fed. Cir. 1983). Claims 18-26 are allowable.

For the reasons stated above all claims are allowable over this rejection.

2. Whether claim 24 is unpatentable over Sun Microsystems (EP 1126380) in view of Bellwood et al. (US 6401132) and further in view of Povilus (US 5740425).

Claim 24 is dependent upon independent claim 18 and is allowable as being dependent upon an allowable base claim.

VIII. Claims Appendix

1. A system for converting data in a first hierarchical data scheme into a second hierarchical data scheme, comprising:

a template defining the second hierarchical data scheme, wherein a hierarchical data scheme is a scheme that groups data and its context;
a dynamic data generation module contained in the template; and
a data source, in communication with the dynamic data generation module, containing data in the first hierarchical data scheme.

2. The system of claim 1, wherein the template and the dynamic data generation module are contained in a server.

3. The system of claim 2, further including a driver connected between the dynamic data generation module and the data source.

4. The system of claim 3, further including a developer module contained in the server for creating the dynamic data generation module.

5. The system of claim 1, wherein the template is a static extensible markup language document.

6. The system of claim 1, wherein the template is an extensible markup language document type definition.

7. The system of claim 1, wherein the template is an extensible markup language schema.

8. The system of claim 1, wherein the first hierarchical data scheme is selected from the group of: extensible markup language schemes, relational databases, non-relational databases, extensible markup language databases and self describing databases.

9. The system of claim 1, wherein the second hierarchical data scheme is selected from the group of: extensible markup language schemes, relational databases, non-relational databases, extensible markup language databases and self describing databases.

10. The system of claim 1, wherein the dynamic data generation module includes a query directed to the data source.

11. The system of claim 1, wherein the dynamic data generation module includes a data mapping between the first hierarchical data scheme and the second hierarchical data scheme.

12. The system of claim 4, wherein the developer module contains a wizard that walks a user through a process of creating the dynamic data generation module.

13. A method of converting data in a first hierarchical data scheme into a second hierarchical data scheme, comprising the steps of:

- a) publishing a dynamic template in a server;
- b) receiving an instruction from a client at the dynamic template;
- c) executing the dynamic template; and
- d) when a dynamic data generation module is executed, performing a data transfer operation that converts data in the first hierarchical data scheme into the second hierarchical data scheme, wherein a hierarchical data scheme is a scheme that groups data and its context.

14. The method of claim 13, wherein step (a) further includes the steps of:
- a1) receiving a template;
 - a2) determining for each element of the template if a dynamically generated data is required;
 - a3) when the dynamically generated data is required, receiving a data source for obtaining the dynamically generated data.
15. The method of claim 14, further including the steps of:
- a4) receiving a data mapping between the first hierarchical data scheme and the second hierarchical data scheme.
16. The method of claim 15 wherein step (a4) further includes the steps of:
- i) when the first hierarchical data scheme is a non-extensible markup language and the second hierarchical data scheme is a second non-extensible markup language, creating a first data mapping between the first hierarchical data scheme and an intermediate extensible markup scheme;
 - ii) creating a second data mapping between the intermediate extensible markup scheme and the second hierarchical data scheme.
17. The method of claim 15, further including the step of'
- a5) receiving a key associated with the data mapping.

18. A method of converting data in a hierarchical data scheme into an extensible markup language scheme, comprising the steps of:

- a) receiving a static extensible markup language template;
- b) determining for each element of the static extensible markup language template if a datum needs to be dynamically generated;
- c) when the datum needs to be dynamically generated, receiving a data source having data in the hierarchical data scheme for acquiring the datum;
- d) receiving a data map between a data element in the data source and a metatag in the static extensible markup language template; and
- e) repeating steps (b) through (d) for every element of the static extensible markup language template to form a dynamic data conversion program.

19. The method of claim 18, wherein step (a) further includes the step of receiving a template selected from the group including: an extensible markup language document type definition and an extensible markup language schema.

20. The method of claim 18, wherein step (a) further includes the step of:

- a1) defining an input parameter.

21. The method of claim 18, wherein step (c) further includes the step of:

- c1) receiving a driver.

22. The method of claim 18, wherein step (c) further includes the step of:

- c1) generating a query to the data source.

23. The method of claim 18, wherein step (d) further includes the step of:

d1) receiving a screen having a list of elements from the data source and a list of metatags from the static extensible markup language template.

24. The method of claim 18, wherein step (c) further includes the step of:

c1) displaying an incomplete version of a dynamic extensible markup language template wherein a static element is shown in a first color and a dynamic element is shown in a second color.

25. The method of claim 18, further including the steps of:

e) publishing the dynamic data conversion program to a server;

f) when a query is received at the server for the dynamic data conversion program, executing the dynamic data conversion program to form an extensible markup language document.

26. A method of converting data in an extensible markup language scheme into a hierarchical data scheme, comprising the steps of:

a) receiving a sample extensible markup language file;

b) determining for each element of the sample extensible markup language file if a datum needs to be dynamically processed;

c) when the datum needs to be dynamically processed, receiving an extensible markup language element location for acquiring the datum;

d) receiving a data map between a metatag in the sample extensible markup language file and an element of the hierarchical data scheme; and

e) repeating steps (b) through (d) for every element of the sample extensible markup file to form a dynamic data conversion program.

27. The method of claim 26, wherein step (a) further includes the step of:

a1) defining a key.

28. The method of claim 26, wherein step (d) further includes the steps of:

d1) receiving a query type;

d2) generating a query.

29. The method of claim 28, wherein step (d1) further includes receiving an insert query type.

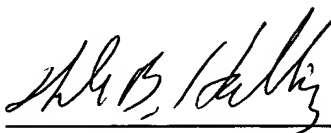
IX. Evidence Appendix

None

X. Related Proceedings Appendix

None

Respectfully submitted,
(Vandersluis)

By: 
Attorney for the Applicant
Dale B. Halling
Registration No. 38,170
Phone: (719) 447-1990
Fax: (719) 447-9815